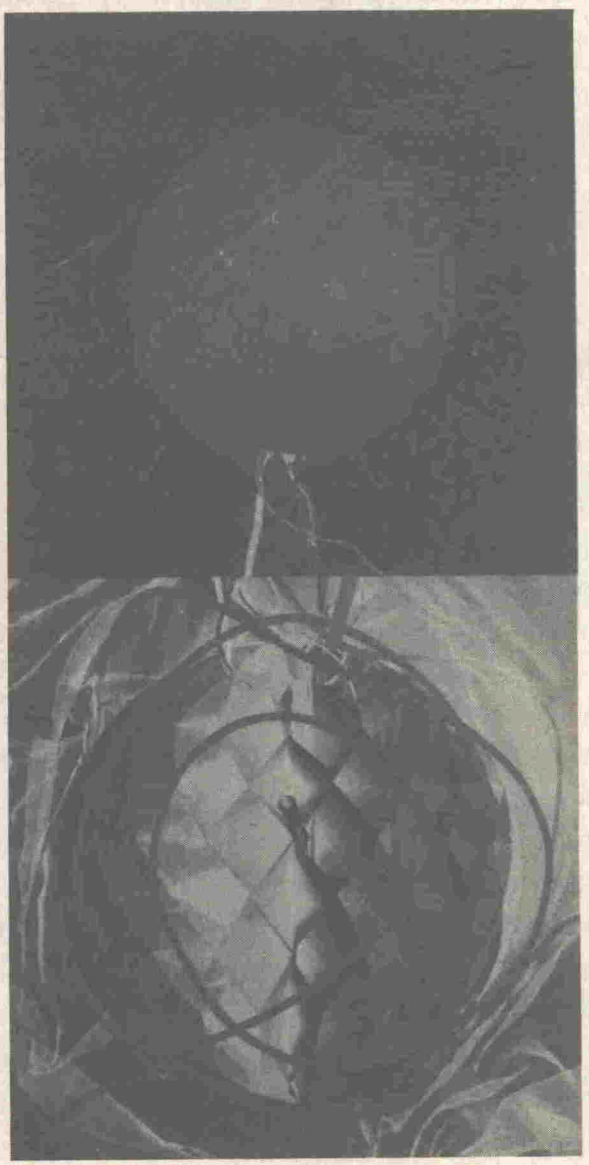
จุฬารัตน์ - ธิติพงศ์ ตันประเสริฐ

ทฤษฎีและการประยุกต์ใช้ โครงข่ายประสาทเทียม (ตอนที่ 1)

เป็นแบบจำลองในยุคแรก ของสาขา ANN เพื่อเสนอให้เห็น แนวทางการแก้ปัญหาแบบใหม่ ด้วยคอมพิวเตอร์



Artificial Neural Network (ANN) หรือ Neural Network หรือ Neural Net หรือโครงข่ายประสาทเทียม ล้วนเป็นคำที่มีความหมายเหมือนกันเป็นชื่อของแขนงหนึ่งในสาขาวิทยาการคอมพิวเตอร์ ในการพยายามที่จะเลียนแบบการทำงานของสมองมนุษย์ เฉพาะงานที่คอมพิวเตอร์ยังทำไม่ได้หรือทำได้ไม่ดี อาทิเช่น การอ่านหนังสือ การจัดหมวดหมู่สิ่งของ เป็นต้น ซึ่งในปัจจุบันนี้นักวิทยาศาสตร์ทางสาขาชีววิทยาและสาขาที่เกี่ยวข้องกับการศึกษารูปแบบการทำงานของสมองมนุษย์ก็ยังไม่สามารถระบุรายละเอียดทั้งหมดของกลไกการทำงานของสมองมนุษย์ได้ ส่งผลให้นักวิชาการทางวิทยาการคอมพิวเตอร์เสนอรูปแบบและวิธีการจำลองการทำงานของสมองมนุษย์ออกมามากมาย โดยมีเป้าหมายที่ความสามารถของแบบจำลองให้เหมือนการทำงานด้วยมนุษย์มากที่สุด

ก่อนอื่นเรามาดูองค์ประกอบของสมองมนุษย์กันก่อน ในสมองมนุษย์นั้นประกอบไปด้วยเซลประสาท หรือนิวรอน ประมาณ 10^{11} ตัว โดยที่เซลประสาทหนึ่ง ๆ มีเส้นเชื่อมโยงต่อไปยังเซลประสาทอื่น ๆ ประมาณ 10^{4} ตัว และเวลาในการสับเปลี่ยนข้อมูลระหว่างนิวรอนอยู่ในลำดับของ 10^{-3} วินาที ซึ่งนับว่าช้ามากเมื่อเทียบกับเวลาในการ switch ของคอมพิวเตอร์เพียง 10^{-10} วินาทีเท่านั้น อย่างไรก็ตาม สมองมนุษย์สามารถประมวลผลได้รวดเร็วมาก ตัวอย่างเช่น คนเราสามารถจำหน้าแม่ของตนเองได้ภายในเวลาเพียง 10^{-1} วินาที ซึ่งน่าจะต้องผ่านการทำงานเป็นร้อย ๆ ขั้นตอนที่เดียวกว่าที่จะได้คำตอบว่านี่คือหน้าของแม่เรา และเมื่อเทียบกับเวลาในการทำงานของแต่ละนิวรอนแล้ว จึงเกิดการตั้ง

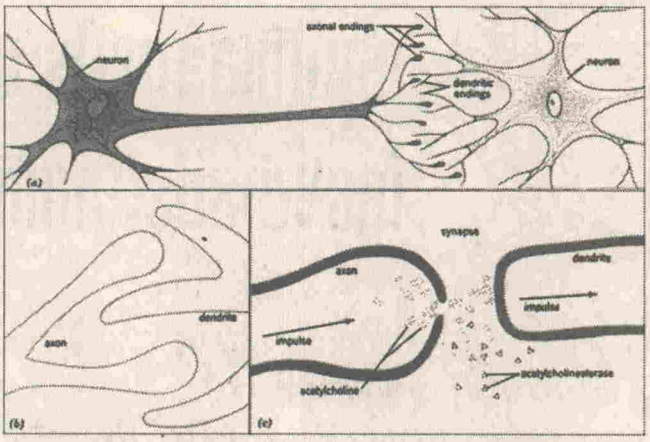
สมมติฐานขึ้นว่า การทำงานของนิวรอนในสมองมนุษย์นั้นน่าจะใช้การประมวลผลแบบขนานขั้นสูง (Highly Parallel Processes) ดังนั้นการที่จะจำลองเซลประสาทและการต่อเชื่อมโยงทั้งหมดของสมองมนุษย์คงจะต้องใช้คอมพิวเตอร์ขนาดใหญ่และสมรรถนะสูงมาก และน่าจะต้องมีการนำการคำนวณแบบขนานเข้ามาร่วมด้วย

จากประวัติศาสตร์พอจะสรุปได้ว่า นักวิจัยของ ANN แบ่งได้เป็น 2 กลุ่มหลัก ๆ คือ กลุ่มหนึ่งมีเป้าหมายที่จะใช้ ANN ในการศึกษาและจำลองขบวนการประมวลผลของสมองสิ่งมีชีวิต แต่อีกกลุ่มหนึ่งมีเป้าหมายที่จะพัฒนาขบวนการ machine learning โดยไม่สนใจว่าขบวนการเหล่านั้นจะเลียนแบบหรือคล้ายกับการทำงานของสมองของสิ่งมีชีวิตหรือไม่ ผู้เขียนทั้งสองไม่มีความรู้ในแง่สมองของสิ่งมีชีวิตมากนัก และขอมอบหน้าที่สำหรับการค้นหาแบบจำลองการทำงานของสมองสิ่งมีชีวิตให้กับนักชีววิทยาและนักวิจัยของ ANN กลุ่มแรก เนื้อหาในบทความตอนนี้และตอนต่อ ๆ ไปเกี่ยวกับเรื่องของ ANN นั้นจะเป็นงานวิจัยของนักวิจัยกลุ่มที่สอง คือเน้นการพัฒนาขบวนการเรียนรู้ของเครื่อง (machine learning)

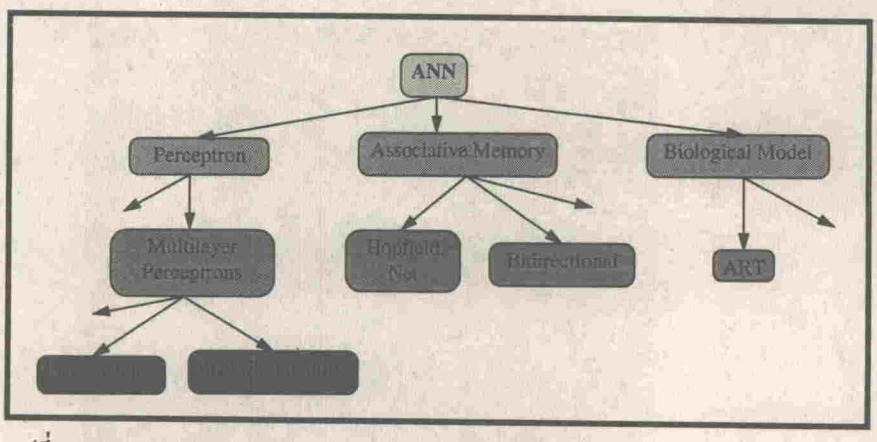
การนำ ANN มาประยุกต์ใช้งานนั้นแบ่งเป็น 2 ขั้นตอนหลัก คือขั้นการฝึกหัด (Training) หรือเรียนรู้ (Learning) และขั้นการทดสอบ (Testing) หรือใช้งานจริง ซึ่งในขั้นตอนการฝึกหัดนั้นมีการพัฒนาและเสนอรูปแบบจำลองการเรียนรู้ออกมามากมาย ซึ่งพอจะแบ่งออกได้เป็น 2 กลุ่มหลัก คือการเรียนรู้แบบครูสอน (Supervised Learning Algorithm) และการเรียนรู้แบบไม่มีครูสอน (Unsupervised Learning Algorithm) ตัวอย่างงานประยุกต์ที่เหมาะสมกับการเรียนรู้แต่ละแบบ อาทิเช่น ถ้าเราต้องการให้ระบบคอมพิวเตอร์ระบุผู้พูดเมื่ออินพุตเป็นเสียงพูดเสียงหนึ่ง ในขั้นการฝึกหัดเราก็ต้องเตรียมข้อมูลคู่อินพุตและคู่เอาต์พุตของแต่ละเสียงพูดไว้ นั่นคือเก็บแฟ้มเสียงพูดเป็นอินพุตและเอาต์พุตก็เป็นชื่อหรือรหัสของผู้พูดคนนั้น ๆ คู่อินพุต-เอาต์พุตทั้งหมดจะถูกส่งเข้าไปใน ANN เพื่อทำการเรียนรู้ ซึ่งงานประยุกต์นี้เหมาะที่จะเลือกใช้การเรียนรู้แบบมีครูสอน เพราะเราสอนให้ ANN ทราบว่าอินพุตไหนเอาต์พุตควรเป็นอะไร แต่ถ้าเปลี่ยนเป็นงานประยุกต์ที่ว่ามีกลุ่มคนอยู่ 10,000 คน ซึ่งเราต้องการแบ่งออกเป็นกลุ่มย่อย ๆ โดยที่เราก็ไม่ทราบว่าจะแบ่งด้วยกฎเกณฑ์อะไรดี ปัญหาอย่างนี้เหมาะกับการเรียนรู้แบบไม่มีครูสอนเป็นอย่างยิ่ง นั่นคือเรามีเฉพาะค่าอินพุตเท่านั้น ส่วนเอาต์พุต ANN จะต้องเป็นผู้ระบุให้เรา อย่างนี้เป็นต้น

อย่างไรก็ตาม ในปัจจุบันนี้มีแบบจำลองของ ANN เยอะมากทั้งที่เป็นแบบมีครูสอนและไม่มีครูสอน นอกจากนี้ยังมีบางรูปแบบที่ผสมผสานทั้งสองรูปแบบเข้ามาใช้ด้วยกันก็มี แบบจำลองที่ได้รับความนิยมค่อนข้างมากแบ่งออกได้เป็น 3 กลุ่มใหญ่ ๆ คือ Perceptron, Associative Memory, และ Biological Model พอแสดงเป็นลำดับได้ดังแสดงในรูปที่ 2

สำหรับในบทความตอนแรกนี้จะบรรยายเรื่องของ Perceptron เป็นประเดิม เพราะเป็นโครงสร้างพื้นฐานของสาขา ANN สามารถประยุกต์ใช้กับงานง่าย ๆ ได้ และแนะนำ ADALINE ซึ่งเป็นแบบจำลองที่นำมาใช้งานจริงกันมากพอสมควร สำหรับเรื่องอื่น ๆ จะได้นำเสนอในตอนต่อ ๆ ไป ANN เป็นรูปแบบในการแก้ปัญหาอีกแบบหนึ่ง ANN จะสร้าง



รูปที่ 1 แสดงส่วนประกอบของเซลประสาทสมองมนุษย์
(ภาพจาก http://www.word.com/newstaff/interns/beta4/biology/dendrites.html)



รูปที่ 2 แสดงแบบจำลองหลัก ๆ ของ ANN [3]

ขบวนการ (Process) ขึ้นเองจากความรู้ ซึ่งได้มาโดยการเรียนรู้จากตัวอย่าง (Learning by Examples) ทั้งนี้ถ้าตัวอย่างที่นำมาสอนแก่โครงข่ายประสาทเทียมมีจำนวนมากพอและวิธีการสอนดีพอ ขบวนการที่ถูกสร้างขึ้นก็จะมีความถูกต้องเพียงพอสำหรับการนำไปใช้งานจริงได้

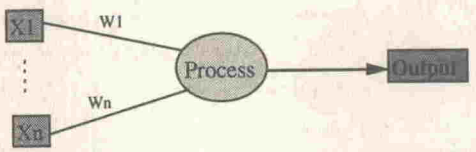
ก่อนอื่นเรามาดูการเลียนแบบการสร้างเซลประสาทหรือที่เรียกกันว่านิวรอน ในโครงข่ายประสาทเทียมเทียบกับเซลประสาทในสมองมนุษย์จริงซึ่งแสดงในรูปที่ 1 (ภาพบน) เห็นได้ว่านิวรอนของสมองประกอบไปด้วย Nucleus, Dendrite และ Axon ซึ่งนิวรอนแต่ละตัวจะรับข้อมูลผ่านทาง Dendrites ทำการประมวลผลใน Nucleus และส่งเอาต์พุตให้นิวรอนอื่น ๆ ผ่านทาง Axon ฟังก์ชันเหล่านี้ถูกจำลองมาเป็นนิวรอนใน ANN ดังแสดงในรูปที่ 3 มีส่วนที่รับ input ซึ่งเทียบได้กับส่วน dendrite การประมวลผลในช่วงกลางก็เทียบได้กับการทำงานของ nucleus และส่วนส่ง output เทียบเท่ากับ axon นั่นเอง

คุณสมบัติและความสามารถที่เป็นประโยชน์ของ ANN

1. มีความยืดหยุ่นสูงจนสามารถจำลองขบวนการของปัญหาใด ๆ ได้
2. มีความสามารถในการจำชุดของคู่อินพุต-เอาต์พุตที่มีความซับซ้อนมากจนไม่สามารถจำลองแบบในเชิงความน่าจะเป็นได้
3. มีความสามารถในการปรับตัวเข้ากับการเปลี่ยนแปลงของสิ่งแวดล้อม
4. มีความสามารถในการตอบสนองต่อข้อมูลที่ไม่เคยเห็น
5. ความรู้จะกระจายอยู่ทั่วทั้งโครงสร้างของโครงข่ายประสาทเทียม ทำให้ดำเนินการกับสารสนเทศเชิงอรรถาธิบาย (Contextual Information) ได้อย่างเป็นธรรมชาติ

ลักษณะของปัญหาที่เหมาะสมกับ ANN

1. ตัวอย่างข้อมูลสำหรับการฝึกหัดประกอบด้วย attribute หลาย ๆ ค่า
2. เอาต์พุตที่ต้องการสามารถเป็นค่าจำนวนเต็ม หรือจำนวนจริง หรือเป็นเวกเตอร์ของจำนวนเต็มหรือจำนวนจริงก็ได้
3. ตัวอย่างข้อมูลสามารถมีความผิดพลาดปะปนอยู่ได้
4. ไม่มีปัญหากับขั้นตอนการฝึกหัดซึ่งอาจใช้เวลานานมาก
5. ต้องการการประเมินผลเอาต์พุตอย่างรวดเร็ว
6. ไม่ต้องการเข้าใจวิธีการแก้ปัญหา ต้องการให้ปัญหาได้รับการแก้ไขเท่านั้น

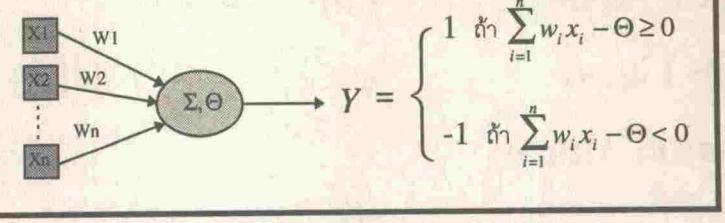


รูปที่ 3
แสดงแบบจำลองของนิวรอนใน ANN

แบบจำลอง McCulloch-Pitts

ANN จะประกอบด้วยนิวรอนจำนวนหนึ่งต่อเข้าด้วยกันโดยเอาต์พุตจากนิวรอนหนึ่งอาจส่งต่อไปให้นิวรอนตัวอื่น ๆ ได้อีกหลายตัว แล้วแต่โครงสร้างของโครงข่ายประสาทเทียมจะถูกออกแบบมาในแต่ละการใช้งาน แบบจำลองของนิวรอนแบบหนึ่งชื่อว่าแบบจำลอง McCulloch-Pitts ซึ่งถูกเสนอโดย McCulloch และ Pitts ตั้งแต่ปี ค.ศ. 1943 มีลักษณะดังแสดงไว้ในรูปที่ 4

ค่าของการรวมเชิงเส้น (linear combination) ระหว่างอินพุตและค่าน้ำหนักจะถูกคำนวณและนำไปเปรียบเทียบกับค่าระดับ (threshold : Θ) เพื่อพิจารณาว่าเอาต์พุตจะออกเป็น 1 หรือ -1 ซึ่งไฮเปอร์เพลนของการตัดสินใจนี้คือ $\sum_{i=1}^{n} w_i x_i - \Theta = 0$ นั่นเอง



รูปที่ 4 แสดงแบบจำลอง McCulloch-Pitts

จากแบบจำลองในรูปที่ 4 สมมติว่ามีอินพุตเพียง 2 ค่าคือ (I1 และ I2) จะได้ว่าค่าเอาต์พุตของนิวรอน Y อาจกำหนดได้เป็น

$Y = 1$ เมื่อ $w_{1}I_{1} + w_{2}I_{2} \geq \Theta$
หรือ $w_{1}I_{1} + w_{2}I_{2} - \Theta \geq 0$
$Y = -1$ เมื่อ $w_{1}I_{1} + w_{2}I_{2} < \Theta$
หรือ $w_{1}I_{1} + w_{2}I_{2} - \Theta < 0$

โดยที่ $w_i, i = 1, 2$ เรียกว่าค่าน้ำหนัก (Weights) เป็นค่าที่ใช้ในการขยายหรือลดหรือกลับขั้วอินพุต I_i, $i = 1, 2$ และ Θ เรียกว่าค่าระดับ (threshold) หลักการในการใช้นิวรอนในการเรียนรู้ปัญหาก็คือการปรับค่าน้ำหนักและค่าระดับตามข้อมูลตัวอย่างนั่นเอง วิธีการทั่วไปในการสอนให้นิวรอนในรูปที่ 4 เรียนรู้จนจำลองวิธีการให้ผลลัพธ์จนถูกต้อง สามารถเขียนเป็นอัลกอริธึมได้ดังนี้

อัลกอริธึม

1. ตั้งค่าเริ่มต้นน้ำหนัก (โดยปกติจะใช้วิธีสุ่มค่าให้มีค่าน้อยๆ มักใช้ระหว่าง [-1, 1])

2. สำหรับแต่ละชุดตัวอย่าง [I1 I2 ... In] : [t] โดย [I1 I2 ... In] คืออินพุตเวกเตอร์ และ [t] คือค่าเป้าหมาย (target) ทำ

2.1 ส่งตัวอย่างเข้าทางอินพุต

2.2 คำนวณค่าเอาต์พุต y โดยคำนวณค่าของ $\sum_{i=1}^{n} W_i I_i - \Theta$ แล้วเปรียบเทียบกับ 0

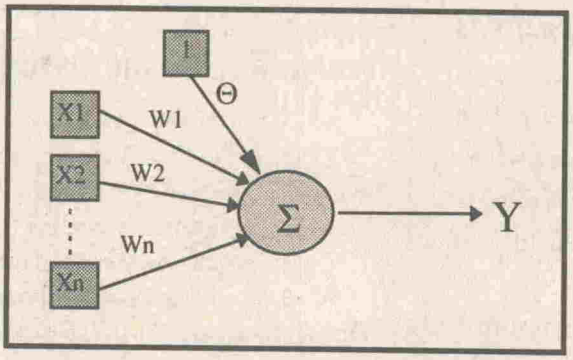
2.3 ถ้า $y \neq t$ ปรับค่าน้ำหนักตามแต่กฎการเรียนรู้จะกำหนด

3. ถ้านิวรอนให้เอาต์พุตตรงกับเป้าหมาย (target) t สำหรับทุกชุดของตัวอย่าง จบ มิฉะนั้น กลับไปข้อ 2

แบบจำลอง McCulloch-Pitts เป็นโครงสร้างพื้นฐานแบบแรกของนิวรอนเทียม มีการใช้ค่าน้ำหนักและค่าระดับที่ตายตัว และใช้ฟังก์ชันลำดับขั้นเป็นตัวกำหนดค่าเอาต์พุตที่จะส่งต่อไปยังนิวรอนอื่น

แบบจำลองเพอร์เซพตรอนเชิงเส้นอย่างง่าย (Simple Linear Perceptron Model)

ในหัวข้อนี้จะเสนอชนิดของ ANN แบบหนึ่งซึ่งมีสถาปัตยกรรมที่ใช้แบบจำลอง McCulloch-Pitts เป็นพื้นฐาน และมีกฎการเรียนรู้ (learning rule) โดยเฉพาะเรียกว่าเพอร์เซพตรอนเชิงเส้น (Linearr Perceptron) สถาปัตยกรรมของเพอร์เซพตรอนเชิงเส้นแบบง่ายอาจกล่าวได้ว่าเป็นอันเดียวกับแบบจำลอง McCulloch-Pitts ความแตกต่างก็คือการที่ค่าระดับจะถูกกำหนดมาเลยว่าเป็นชนิดเปลี่ยนค่าได้เช่นเดียวกับค่าน้ำหนักซึ่งอาจแสดงได้ดังรูปที่ 5



รูปที่ 5 แสดงสถาปัตยกรรมของเพอร์เซพตรอนเชิงเส้นอย่างง่าย

เราสามารถเขียนค่าน้ำหนักและค่าอินพุตได้ในรูปของเวกเตอร์ดังนี้

$$W = [w_1, w_2, ..., w_n, \Theta]^T$$
$$X = [x_1, x_2, ..., x_n, 1]^T$$

จากนั้นกำหนดให้อินพุตลัพธ์ (net input) มีค่าเป็น $U = W^T X$

ดังนั้นเราจะได้ว่าเอาต์พุตคือ

$$Y = \begin{cases} 1, & u \geq 0 \\ -1, & u < 0 \end{cases}$$

จริง ๆ แล้วเราสามารถกำหนดให้ค่าเอาต์พุตอยู่ในช่วง [0,1] ก็ได้ แต่โดยทั่วไปการใช้ช่วง [-1,1] จะให้ผลที่ดีกว่า

อัลกอริธึมของเพอร์เซพตรอนเชิงเส้น

1. ตั้งค่าเริ่มต้นน้ำหนักโดยวิธีสุ่ม

2. สำหรับแต่ละชุดตัวอย่าง **I : T** (โดย **I** คือเวกเตอร์มิติ **n**) ทำ

2.1 **X = I**

2.2 คำนวณอินพุตลัพธ์ $U = W^T X$

2.3 คำนวณการกระตุ้น (Activation) ที่เอาต์พุต

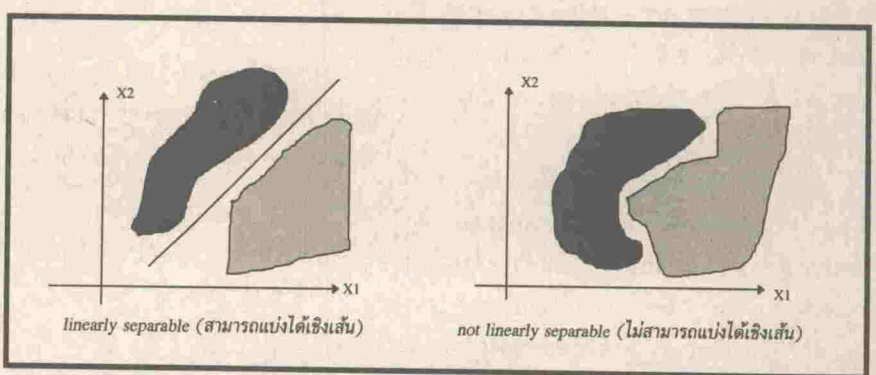
$$y = f(u) = \begin{cases} 1, & u \geq 0 \\ -1, & u < 0 \end{cases}$$

2.4 ถ้า $y \neq t$ ปรับค่าน้ำหนักโดยกฎการเรียนรู้ของเพอร์เซพตรอนเชิงเส้น

$$W(\text{ใหม่}) = W(\text{เก่า}) + \eta t X$$

โดย η เรียกว่าอัตราการเรียนรู้มีค่าในช่วง (0,1]

3. ถ้าค่าน้ำหนักไม่เปลี่ยนแปลงต่อไปและ/หรือค่ารวมของความผิดพลาดกำลังสอง (squared errors : $(t-y)^2$) มีค่าต่ำกว่าค่าที่กำหนดซึ่งโดยปกติจะตั้งให้มีค่าน้อยว่า 0.1 จบมิฉะนั้น กลับไปข้อ 2



รูปที่ 6 แสดงปัญหาที่เป็นแบบ Linearly Separable และ Not Linearly Separable.

มีทฤษฎีที่กล่าวไว้ว่า "อัลกอริธึมของเพอร์เซพตรอนเชิงเส้นจะเปลี่ยนค่าน้ำหนักจนได้ค่าที่ให้แบ่งกลุ่มตัวอย่างที่นำมาฝึกได้อย่างถูกต้อง ถ้ากลุ่มของตัวอย่างที่นำมาฝึกฝนนั้นสามารถแบ่งเป็นสองกลุ่มได้เชิงเส้น (Linearly separable)" ดังแสดงในรูปที่ 6 ซึ่งแสดงลักษณะของปัญหาที่แยกได้เชิงเส้น และแบบแยกเชิงเส้นไม่ได้

ตัวอย่าง การใช้ Perceptron แก้ปัญหาตรรก "และ" (AND)

I1	I2	t
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

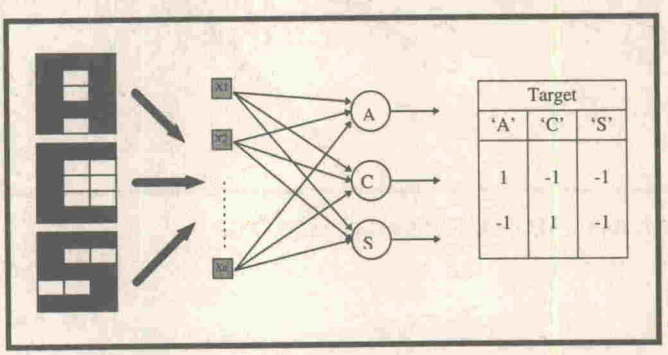
สมมติให้ค่าน้ำหนักเริ่มต้นเป็น 0 รวมทั้งค่าระดับด้วยนั่นด้วย W = [0 0 0] และเราทดลองที่ค่าอัตราการเรียนรู้ $\eta = 1$ (ตารางที่ 1)

ต่อจากนี้จะพบได้ว่าค่าน้ำหนักไม่เปลี่ยนอีกต่อไป และเอาต์พุต y ก็ถูกต้องทุกกรณีแล้ว นั่นคือ Perceptron เรียนรู้ตรรก "AND" แล้ว โดยได้ค่าน้ำหนัก $w_1=1$, $w_2=1$, $\Theta = -1$ จาก Perceptron ขั้นพื้นฐานดังกล่าวเราสามารถขยายแนวคิดออกไปสำหรับกรณีที่คำตอบมีค่าที่เป็นไปได้มากกว่าสองค่าได้ โดยใช้นิวรอนมากกว่าหนึ่งตัว ตัวอย่างเช่น การขยายแนวคิดของ Perceptron สำหรับปัญหาการรู้จำตัวอักษรสามารถทำได้ดังแสดงในรูปที่ 7 ซึ่งเป็นการรู้จำตัวอักษรภาษาอังกฤษ 3 ตัว

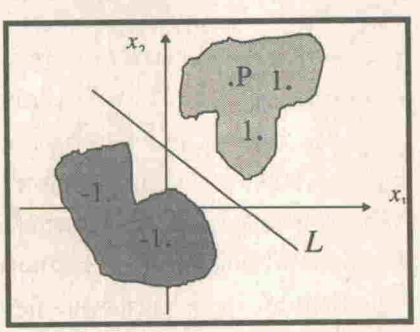
ตัวอักษร 'A', 'C', 'S' เป็นภาพขนาด 5x3 จุด โดยใช้เพอร์เซพตรอนขนาดอินพุต 15 บิต และใช้นิวรอนสามตัว โดยกำหนดเป้าหมายของคำตอบดังนี้ นิวรอน $\iota ; \iota \in \{A,C,S\}$ มีค่า = 1 ถ้าอินพุตคือ อักษร ι และ -1 ถ้าไม่ใช่ โดยทั่วไปแล้ว นอกเหนือไปจากความสามารถในการจำตัวอักษรที่สอนแล้ว โครงข่ายประสาทเทียมยังสามารถให้ผลคำตอบที่ถูกต้องต่ออินพุตที่ผิดเพี้ยนไปบ้าง เช่น

ตารางที่ 1

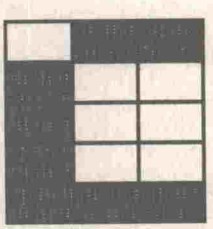
	[x1	x2	1]	u	y	t	[w1	w2	Θ]
	-	-	-	-	-	-	[0	0	0]
รอบที่ 1	[1	1	1]	0	1	1	[0	0	0]
	[1	-1	1]	0	1	-1	[-1	1	-1]
	[-1	1	1]	1	1	-1	[0	0	-2]
	[-1	-1	1]	-2	-1	-1	[0	0	-2]
รอบที่ 2	[1	1	1]	-2	-1	1	[1	1	-1]
	[1	-1	1]	-1	-1	-1	[1	1	-1]
	[-1	1	1]	-1	-1	-1	[1	1	-1]
	[-1	-1	1]	-3	-1	-1	[1	1	-1]
รอบที่ 3	[1	1	1]	1	1	1	[1	1	-1]



รูปที่ 7 แสดงการใช้เพอร์เซพตรอนเชิงเส้นในการแก้ปัญหาการรู้จำตัวอักษร

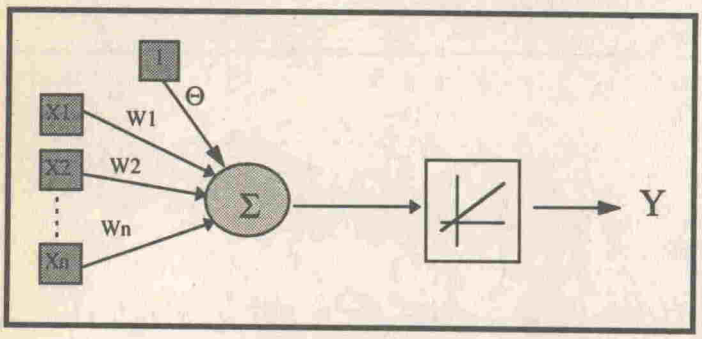


รูปที่ 8 แสดงความสามารถในการจัดหมวดหมู่ของ ANN

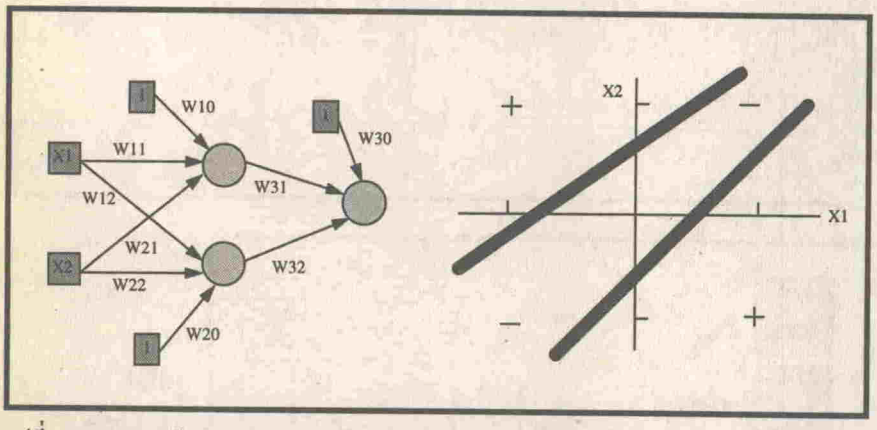


ได้ค่อนข้างถูกต้องอีกด้วย ซึ่งสามารถอธิบายเชิงเปรียบเทียบได้ดังรูปที่ 8

ลองดูตัวอย่างในกรณีที่อินพุตเป็น 2 มิติ ดังในรูปที่ 8 ซึ่งเส้นตรง L คือไฮเปอร์เพลนเชิงเส้นที่ได้จากการเรียนรู้จุดสี่จุดที่มีค่าผลตอบเป็น 1 และ -1 ในรูป จะเห็นได้ว่าผลพลอยได้คือ ANN จะให้คำตอบที่ถูกต้องแก่จุด P ได้ด้วย คือจะให้ผลตอบเป็น 1 ทั้งที่ไม่เคยถูกฝึกจุด P มาก่อน ดังนั้นถ้าจุดที่ใช้ทดสอบนั้นไม่ผิดเพี้ยนมากเกินไปจากตัวอย่างที่เราใช้ฝึก ANN ก็สามารถให้ผลตอบที่ถูกต้องได้ด้วย



รูปที่ 9 แสดงสถาปัตยกรรมของโครงข่าย ADALINE



รูปที่ 10 แสดงสถาปัตยกรรมแบบ MADALINE และตัวอย่างปัญหา XOR

โครงข่าย ADALINE กับการเรียนรู้แบบ Widrow-Hoff

เป็น ANN ที่มีสถาปัตยกรรมของโครงข่ายเหมือนกับ Perceptron สิ่งเดียวที่แตกต่างกันคือใน ADALINE ใช้ฟังก์ชันเส้นตรง (linear function) ในการระบุเอาต์พุต ส่วนของ Perceptron ใช้ฟังก์ชันแบบ Hard-Limiting นั่นคือออกเป็นค่า binary (1, -1) โดยได้จากการนำไปเปรียบเทียบกับค่าระดับ อย่างไรก็ตาม โครงข่ายทั้งสองแบบยังคงสามารถใช้แก้ปัญหาได้เฉพาะแบบ Linearly Separable เท่านั้น โครงข่าย ADALINE (ADAptive LInear NEuron) ใช้กับการเรียนรู้แบบ LMS (Least Mean Squares) หรือบางครั้งเรียกว่ากฎของ Widrow-Hoff ซึ่งแตกต่างจากการเรียนรู้ของ Perceptron ตรงจุดที่ว่าการเรียนรู้แบบ Perceptron จะไวต่อค่าความผิดพลาด เส้นแบ่งกลุ่มที่ได้มักจะอยู่ใกล้กับกลุ่มตัวอย่างที่ใช้ในการฝึกหัด แต่ LMS นั้นทำการลดค่าความผิดพลาดให้น้อยที่สุดโดยจะพยายามหาเส้นแบ่งกลุ่มให้ห่างจากกลุ่มตัวอย่างมากที่สุดเท่าที่จะทำได้ ส่งผลให้โครงข่าย ADALINE พร้อมการเรียนรู้แบบ LMS มีความทนทานต่ออินพุตที่มี noise ปะปนมาได้ดีกว่า Perceptron

โครงข่าย ADALINE ได้รับความนิยมในการนำไปใช้งานกับการประมวลผลสัญญาณ (Signal Processing Applications) และโครงข่ายนี้ยังเป็นพื้นฐานของโครงข่ายแบบ Multilayer Network กับการเรียนรู้แบบ Backpropagation ซึ่งได้รับความนิยมสูงสุดในการนำไปใช้งานอีกด้วย ซึ่งผู้เขียนจะอธิบายโครงข่ายแบบหลังนี้ในบทความตอนหน้า

โครงข่ายแบบ ADALINE แสดงได้ดังรูปที่ 9

เราสามารถเขียนค่าน้ำหนักและค่าอินพุตได้ในรูปของเวกเตอร์ดังนี้

$$W = [w_1, w_2, \ldots, w_n, \Theta]^T$$
$$X = [x_1, x_2, \ldots, x_n, 1]^T$$

จากนั้นกำหนดให้อินพุตลัพธ์ (net input) มีค่าเป็น $U = W^T X = Y$ ซึ่งเป็นค่าเอาต์พุตนั่นเอง โดยที่ค่า Θ เป็นค่า bias ซึ่งใช้ในการปรับจุดตัดสมการเส้นตรงในแนวนอนนั่นเอง สำหรับกฎการเรียนรู้จะนำเอาต์พุตที่ได้ไปเปรียบเทียบกับค่า Target Output (t) เพื่อคำนวณค่าความผิดพลาดเพื่อใช้หาค่าน้ำหนักที่ควรปรับเปลี่ยนไป ดังนี้

$E = (t-y)^2$ โดยที่ E คือค่าความผิดพลาด

ในการนี้เราต้องการหาว่าค่าน้ำหนักแต่ละเส้นควรปรับเปลี่ยนไปเท่าไหร่ในเงื่อนไขที่ว่าค่าความผิดพลาดที่เกิดขึ้นมีค่าน้อยที่สุด นั่นคือการหาค่า

$$-\frac{\partial E}{\partial w_t} = 2(t-y)\frac{\partial y}{\partial w_t} = 2(t-y)x_t$$

จะได้ว่าค่าน้ำหนักของแต่ละเส้นเชื่อมที่ถูกปรับไปคือ $\Delta w_t = \eta\,(t-y)\,x_t$

โดยที่ η คืออัตราการเรียนรู้ (0,1] ซึ่งมีข้อสังเกตว่าการเลือกอัตราการเรียนรู้ที่เหมาะสมน่าจะส่งผลให้การเรียนรู้ converge ได้ในเวลาไม่นานเกินไป ถ้าเลือกอัตราการเรียนรู้ใหญ่เกินไป การเรียนรู้อาจจะไม่ converge และถ้าตั้งค่านี้เล็กเกินไปก็จะทำให้การเรียนรู้ช้ามาก ข้อสังเกตในการตั้งค่านี้คือให้มีค่าไม่เกินครึ่งหนึ่งของค่า eigenvalue ของค่าความสัมพันธ์ของเมตริกซ์ R โดยที่

$$R = \frac{1}{n}\sum_{t=1}^{n} x^T{}_t\, x_t$$

อัลกอริธึมของ ADALINE กับการเรียนรู้แบบ LMS

1. ตั้งค่าเริ่มต้นน้ำหนักโดยวิธีสุ่ม และตั้งอัตราการเรียนรู้ η

2. สำหรับแต่ละชุดตัวอย่าง I : T (โดย I คือเวกเตอร์มิติ n) ทำ

2.1) X = I

2.2) คำนวณเอาต์พุต $y = W^T X$

2.3) ถ้า $y \neq t$ ปรับค่าน้ำหนักโดยกฎการเรียนรู้ของ ADALINE

$$W(\text{ใหม่}) = W(\text{เก่า}) + \eta(t-y)W$$

โดย η เรียกว่าอัตราการเรียนรู้มีค่าในช่วง (0,1]

3. ถ้าค่าน้ำหนักไม่เปลี่ยนแปลงต่อไป และ/หรือค่ารวมของความผิดพลาดกำลังสอง (squared errors : $(t-y)^2$) มีค่าต่ำกว่าค่าที่กำหนด ซึ่งโดยปกติจะตั้งให้มีค่าน้อยว่า 0.1 จบมิฉะนั้น กลับไปข้อ 2

นิวรอนแบบ ADALINE สามารถนำมาต่อๆ กันเป็น MADALINE (Many ADALINE) ได้ ซึ่งสามารถนำไปใช้แก้ปัญหาที่ไม่สามารถใช้เส้นตรงเพียงเส้นเดียวแบ่งเขตของตัวอย่างข้อมูลได้ ดังตัวอย่างแสดงในรูปประกอบที่ 10 แสดงโครงข่ายของ MADALINE ที่ประกอบด้วย ADALINE ชั้นซ่อน 2 ตัว ชั้นเอาต์พุต 1 ตัว ซึ่งใช้ในการแก้ปัญหา XOR ได้

ที่เล่ามาข้างต้นนั้นเป็นการแนะนำให้รู้จักกับ ANN แบบพื้นฐาน นั่นคือแบบจำลอง McCulloch-Pitts (นำเสนอโดย McCulloch & Pitts 1943) แบบจำลองเพอร์เซพตรอนเชิงเส้นแบบง่าย (นำเสนอโดย (Rosenblatt 1962), (Minsky & Papert 1969, 1988)) และแบบจำลอง ADALINE (นำเสนอโดย (Widrow & Hoff 1960), (Widrow, Winter & Baxter 1987)) ซึ่งเป็นแบบจำลองในยุคแรกของสาขา ANN เพื่อเสนอให้เห็นแนวทางการแก้ปัญหาแบบใหม่ด้วยคอมพิวเตอร์ ซึ่งไม่เหมือนการการแก้ปัญหาแบบคลาสสิกที่จะต้องมีขั้นตอนอย่างชัดเจน ในบทความหน้าเราจะมาคุยให้ฟังเกี่ยวกับแบบจำลอง Multilayer Perceptrons พร้อมด้วยการเรียนรู้แบบ Backpropagation ซึ่งเป็นที่นิยมของนักวิจัยในการนำไปใช้ในการแก้ปัญหาที่ซับซ้อนต่าง ๆ ในปัจจุบัน

สำหรับตอนนี้ได้ให้อัลกอริธึมของแต่ละแบบจำลองไป เพื่อให้ผู้อ่านได้ทดลองเขียนเป็นโปรแกรมด้วยภาษาที่ผู้อ่านถนัด เพื่อใช้แก้ปัญหาง่าย ๆ ดูก่อน สำหรับท่านที่สนใจอยากจะหาหนังสืออ่านเกี่ยวกับ ANN เพิ่มเติม ผู้เขียนขอแนะนำหนังสือต่างๆ ดังที่แสดงในรายการหนังสืออ้างอิง หรือใน Web Page ต่าง ๆ อาทิเช่น http://www.gc.ssr.upm.es/inves/neural/ann1/anntutorial.html, http://www.shef.ac.uk/psychology/gurney/notes/index.html และ http://www.links.nectec.or.th/thep/study.html มองหาหัวข้อเรื่อง Neural Networks จะพบ link ต่างๆ มากมายเกี่ยวกับ ANN เป็นต้น ในปัจจุบันนี้มีการนำ ANN มาใช้แก้ปัญหาเพื่อใช้งานจริงมากมายแล้ว อาทิเช่น งานการประมวลผลสัญญาณ งานควบคุมเครื่องจักร งานการรู้จำรูปแบบ (Pattern Recognition) ด้านการรักษาผู้ป่วย การสังเคราะห์เสียงพูด การรู้จำเสียงพูด และการทำนายทางธุรกิจ เป็นต้น ติดตามอ่านตัวอย่างการนำ ANN มาประยุกต์ใช้แก้ปัญหาในชีวิตจริงในตอนหน้า แล้วพบกันใหม่ สวัสดี

หนังสืออ้างอิง

1. Martin T. Hagan, Howard B. Demuth, Mark Beale, "Neural Network Design", PWS Pulishing Company, An International Thomson Publishing Company, 1995.

2. Laurene Fausett, "Fundamentals of Neural Networks : Architectures, Algorithms, and Applications", Prentice Hall International Editions, 1994.

3. Robert L. Harvey, "Neural Network Principles", Prentice Hall International Edition, 1994.

4.Simon Haykin, "Neural Networks - A Comprehensive Foundation", Macmillan College Publishing Company, 1994.

5. S. Y. Kung, "Digital Neural Networks", Prentice Hall, 1993.